WAVELET-BASED GRID GENERATION

Leland Jameson[1]
Institute for Computer Applications in Science and Engineering
NASA Langley Research Center
Hampton, VA 23681
email: lmj@icase.edu

ABSTRACT

Wavelets can provide a basis set in which the basis functions are constructed by dilating and translating a fixed function known as the mother wavelet. The mother wavelet can be seen as a high pass filter in the frequency domain. The process of dilating and expanding this high-pass filter can be seen as altering the frequency range that is "passed" or detected. The process of translation moves this high-pass filter throughout the domain, thereby providing a mechanism to detect the frequencies or scales of information at every location. This is exactly the type of information that is needed for effective grid generation. This paper provides motivation to use wavelets for grid generation in addition to providing the final product: source code for wavelet-based grid generation.

[1]This research was supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-19480 while the author was in residence at the Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton, VA 23681.

Contents

1 Introduction **1**

2 Definition of Daubechies-based Wavelets **1**
2.1 The Classical Daubechies Wavelet Theory 1
2.2 Restriction to Finite Dimensions 4

3 From Wavelets to Grid Generation **5**

4 Software Implementation **5**
4.1 Explanation of Subroutines, line by line 5
4.1.1 Generating the Wavelet Coefficients: lines 9-13 6
4.1.2 Generating the Grid: lines 15-43 6
4.1.3 The input and output variables 7
4.2 The Four Subroutines 8
4.2.1 Wavelet Analysis and Select Grid: newgr.f 8
4.2.2 Get Daubechies Wavelet Coefficients: getcoef.f 9
4.2.3 Apply Wavelet Filter: filter.f 10
4.2.4 Apply Boundary Conditions: constext.f 11

5 Doubling the Grid Density **11**

6 2 Dimensional Examples **12**
6.1 The Original WOFD 12
6.2 An Adaptive Spectral Method 12
6.3 Possible Applications to Finite Elements 14

7 Conclusion **15**

List of Figures

1 A domain which contains a high gradient flame front impinging on a Gaussian pulse. 13
2 The grid selected by wavelets for a high gradient region impinging on a pulse. 13

3 Gaussian pulse analyzed with wavelets in an adaptive spectral method. 14
4 An adaptive Chebyshev grid generated by wavelets for an adaptive spectral method. 15

1 Introduction

Wavelet methods are now roughly 10 years old, and there often remains a large gap between a theoretical wavelet paper and the needs of an applied scientist. This paper is an attempt to bridge this gap by providing a short review of wavelet theory followed by a wavelet-based grid generation subroutine. This subroutine has been broken off from a numerical method known at the Wavelet-Optimized Finite Difference (WOFD) method, see [6], and can be used as stand-alone unit. Let us begin by reviewing WOFD.

The logic behind WOFD is as follows: i) If one examines the physical space effect of an adaptive wavelet Galerkin method, one observes a non-uniform grid finite difference method, see [3]. ii) WOFD attemps to mimic this physical-space equivalent of a wavelet Galerkin method by using wavelets to choose a numerical grid and performing the wavelet-equivalent finite difference method on this grid. iii) By performing all calculations in the physical space one avoids problems that wavelet Galerkin methods generally have with boundaries and non-linear terms, see [5], [6].

The grid selection mechanism presented here is very flexible and is independent of the physics which might be associated with a particular problem. Let us begin by defining wavelets in order to give the reader some idea of the reasoning which led to the WOFD grid refinement mechanism.

2 Definition of Daubechies-based Wavelets

This section will introduce wavelets and indicate how the theory which is derived for a continuous independent variable can be reduced to finite dimensions.

2.1 The Classical Daubechies Wavelet Theory

To define Daubechies-based wavelets, see [1] for the original work, consider the two functions $\phi(x)$, the scaling function, and $\psi(x)$, the wavelet. The scaling function is the solution of the dilation equation,

$$\phi(x) = \sqrt{2} \sum_{k=0}^{L-1} h_k \phi(2x - k), \tag{1}$$

where $\phi(x)$ is normalized $\int_{-\infty}^{\infty} \phi(x)dx = 1$, and the wavelet $\psi(x)$ is defined in terms of the scaling function,

$$\psi(x) = \sqrt{2} \sum_{k=0}^{L-1} g_k \phi(2x - k). \tag{2}$$

One builds an orthonormal basis from $\phi(x)$ and $\psi(x)$ by dilating and translating to get the following functions:

$$\phi_k^j(x) = 2^{-\frac{j}{2}} \phi(2^{-j}x - k), \tag{3}$$

and

$$\psi_k^j(x) = 2^{-\frac{j}{2}} \psi(2^{-j}x - k), \tag{4}$$

where $j, k \in Z$. Here j is the dilation parameter and k is the translation parameter. The coefficients $H = \{h_k\}_{k=0}^{L-1}$ and $G = \{g_k\}_{k=0}^{L-1}$ are related by $g_k = (-1)^k h_{L-k}$ for $k = 0, ..., L-1$. All wavelet properties are specified through the parameters H and G. If one's data is defined on a continuous domain such as $f(x)$ where $x \in R$ is a real number then one uses $\phi_k^j(x)$ and $\psi_k^j(x)$ to perform the wavelet analysis. If, on the other hand, one's data is defined on a discrete domain such as $f(i)$ where $i \in Z$ is an integer then the data is analyzed, or filtered, with the coefficients H and G. In either case, the scaling function $\phi(x)$ and its defining coefficients, H, detect localized low frequency information, i.e., they are low-pass filters (LPF), and the wavelet $\psi(x)$ and its defining coefficients G detect localized high frequency information, i.e., they are high-pass filters (HPF). Specifically, H and G are chosen so that dilations and translations of the wavelet, $\psi_k^j(x)$, form an orthonormal basis of $L^2(R)$ and so that $\psi(x)$ has M vanishing moments which determines the accuracy. In other words, $\psi_k^j(x)$ will satisfy

$$\delta_{kl}\delta_{jm} = \int_{-\infty}^{\infty} \psi_k^j(x)\psi_l^m(x)dx, \tag{5}$$

where δ_{kl} is the Kronecker delta function, and the accuracy is specified by requiring that $\psi(x) = \psi_0^0(x)$ satisfy

$$\int_{-\infty}^{\infty} \psi(x)x^m dx = 0, \tag{6}$$

for $m = 0, ..., M-1$. Under the conditions of the previous two equations, for any function $f(x) \in L^2(R)$ there exists a set $\{d_{jk}\}$ such that

$$f(x) = \sum_{j \in Z} \sum_{k \in Z} d_{jk} \psi_k^j(x), \tag{7}$$

where

$$d_{jk} = \int_{-\infty}^{\infty} f(x) \psi_k^j(x) dx. \tag{8}$$

For Daubechies wavelets the number of coefficients in H and G, or the length of the filters H and G, denoted by L, is related to the number of vanishing moments M by $2M = L$. The coefficients H needed to define compactly supported wavelets with a higher degree of regularity can be found in [1]. As is expected, the regularity increases with the support of the wavelet. The usual notation to denote a Daubechies-based wavelet defined by coefficients H of length L is D_L.

It is usual to let the spaces spanned by $\phi_k^j(x)$ and $\psi_k^j(x)$ over the parameter k, with j fixed, be denoted by V_j and W_j respectively,

$$V_j = \underset{k \in Z}{span} \ \phi_k^j(x), \tag{9}$$

$$W_j = \underset{k \in Z}{span} \ \psi_k^j(x). \tag{10}$$

The spaces V_j and W_j are related by,

$$... \subset V_1 \subset V_0 \subset V_{-1} \subset ..., \tag{11}$$

and

$$V_j = V_{j+1} \bigoplus W_{j+1}. \tag{12}$$

The previously stated condition that the wavelets form an orthonormal basis of $L^2(R)$ can now be written as,

$$L^2(R) = \bigoplus_{j \in Z} W_j. \tag{13}$$

2.2 Restriction to Finite Dimensions

Of course, infinite sums are meaningless when one begins to implement wavelet analysis on a computer where there is always a largest scale, a smallest scale, as well as boundaries. That is, one must limit the range of the scale parameter j and the location parameter k. The location parameter k can be limited by, say, imposing periodic boundary conditions which would require that k also be periodic or by building special scaling functions and wavelets at the boundaries.

Consider now the scale parameter j. As stated above, the wavelet expansion is complete in the sense that an arbitrary function with finite energy can be represented by 'summing up' the orthogonal subspaces W_j which contain frequency components related to the parameter j: $L^2(R) = \bigoplus_{j \in Z} W_j$. Therefore, any $f(x) \in L^2(R)$ can be written as,

$$f(x) = \sum_{j \in Z} \sum_{k \in Z} d_k^j \psi_k^j(x).$$

In this expansion, functions with arbitrarily small-scale structures can be represented. In practice, however, there is a limit to how small the smallest structure can be. This would depend, for example, on how fine the grid is in a numerical computation scenario or perhaps what the sampling frequency is in a signal processing scenario. Therefore, on a computer an expansion would take place in a space such as,

$$V_0 = W_1 \oplus W_2 \oplus \ldots \oplus W_J \oplus V_J, \tag{14}$$

and would appear as,

$$P_{V_0} f(x) = \sum_{k \in Z} s_k^J \phi_k^J(x) + \sum_{j=1}^{J} \sum_{k \in Z} d_k^j \psi_k^j(x), \tag{15}$$

where, again, $d_k^j = \int_{-\infty}^{\infty} f(x) \psi_k^j(x)$, and $s_k^J = \int_{-\infty}^{\infty} f(x) \phi_k^J(x)$. In this expansion, scale $j = 0$ is arbitrarily chosen as the finest scale that is needed, and scale J would be the scale at which a kind of local average, $\phi_k^J(x)$, provides sufficient large scale information.

3 From Wavelets to Grid Generation

The idea of using wavelets to generate numerical grids began with the observation in [3] that the essence of an adaptive wavelet-Galerkin method is nothing more than a finite difference method with grid refinement. So, instead of letting the magnitude of wavelet coefficients choose which basis functions to use in a Galerkin approach, let the same coefficients choose which grid points to use and then think of the wavelet method in a collocation sense.

In other words, suppose a calculation begins with N evenly-spaced samples of a function $\vec{f}$ and that some quadrature method produces N scaling function coefficients on the finest scale denoted by V_0. If the spacing between adjacent values in the vector $\vec{f}$ is Δx then this is also the physical-space resolution of any calculation done in V_0. Now, decompose V_0 once to get $V_0 = V_1 \oplus W_1$. Similarly speaking, the physical space resolution of V_1 is $2\Delta x$ and the refinement from the $2\Delta x$ physical-space resolution to the Δx physical-space resolution is dictated by the wavelet coefficients in W_1. This is the reasoning which led to WOFD and to the following subroutine which is at the heart of WOFD. The remainder of the paper is concerned with giving the reader an idea of how the WOFD grid refinement software works.

4 Software Implementation

The philosophy of the this paper is to get the reader to first try the software and once the reader is convinced that it is a reliable grid generator, then move onto the theory. So, let us commence with an examination of the included WOFD grid generator.

The main subroutine *newgr.f* performs two major functions. The lines of code from line 9 to line 13 perform the wavelet analysis producing the wavelet coefficients of the input function *fi*, and from line 15 to line 43 the numerical grid is generated based on these wavelet coefficients.

4.1 Explanation of Subroutines, line by line

This subsection will explain the grid generation subroutines almost line by line.

4.1.1 Generating the Wavelet Coefficients: lines 9-13

To illustrate, assume that the user has determined that 8 is the maximum desired ratio between the maximum Δx and the minimum Δx. As noted above, this corresponds to 3 wavelet decompositions. As above, let V_0 denote the subspace spanned by scaling functions on the finest scale. One wavelet decomposition produces the division of V_0 into V_1 and W_1: $V_0 = W_1 \oplus V_1$. Similarly, three wavelet decompositions produces, $V_0 = W_1 \oplus W_2 \oplus W_3 \oplus V_3$. Each of these wavelet decompositions is performed by the subroutine *filter.f*. The input to *filter.f* is the variable *Extdata* which contains the scaling function coefficients for subspace V_i and the output variables of *filter.f* are the variables *data* and *HPF* which contain the coefficients for the subspaces V_{i+1} and W_{i+1}, respectively. The variable is named *Extdata* because the coefficients of V_i have been 'extended' to reflect the boundary conditions. In this version of the program, the *data* is extended by adding constant scaling function coefficient values to the ends of the vector *data* by the routine *constext.f*. Note, if one desires periodic boundary conditions then one 'wraps' the scaling function coefficients around such that one extends the vector *data* by returning to the beginning of the same vector. Likewise, if one wants a smoother extension of *data* then one can write a routine which extends linearly or by some other higher order polynomial.

4.1.2 Generating the Grid: lines 15-43

The lines of code from 16 to 20 add the grid points which are referred to as the base grid. That is, these points are evenly-spaced and depend only on the number of wavelet decompositions one has chosen. For example, if $N = 128$ and $Nd = 3$ then the base grid will consist of $16 = 128/2^3$ evenly-spaced points. If the input function *fi* is smooth with respect to the threshold, *th*, then the output grid *xo* could very well be composed only of the base grid.

The lines of code from 21 to 42 add the wavelet refinement to the base grid. Using the standard wavelet notation for the example with 3 decompositions, $V_0 = W_1 \oplus W_2 \oplus W_3 \oplus V_3$, the base grid corresponds to the scaling function subspace V_3 in which all the scaling functions are used. If the spacing between grid points in V_0 is Δx then the spacing between grid points in V_3 will be $8\Delta x$. The addition of the wavelets in W_3 which have coefficients larger in magnitude than the threshold *th* will refine the grid to a spacing of $4\Delta x$

in these large coefficient regions. Likewise, adding W_2 refines to $2\Delta x$ and adding W_1 refines to the finest scale of Δx. Testing the magnitude of the wavelet coefficients occurs on line 32 of the code. The variable *iflagpoint* is used with *iw* in order to include a grid point *xi(ipnt)* if, say, 2/3 of the wavelet coefficients in the region around *xi(ipnt)* are large. This mechanism adds a kind of 'softness' to the grid selection mechanism and appears to work very well when *iw = 1*. Line 38 is where the new grid *xo* is constructed. Lines 44 and 45 simply add the right-hand boundary grid point and function value.

4.1.3 The input and output variables

Input Variables

- xi = The evenly-spaced grid point values.
- fi = The evenly-spaced samples of the function which is to be analyzed. fi(1) = value at left-hand boundary. fi(N+1) = value at right-hand boundary. If boundary conditions are periodic, fi(1) = fi(N+1).
- L = Defines which wavelet is used. For Daubechies 4, L=4.
- N = The number of points in *fi* minus 1. N is a power of 2.
- th = Threshold to determine which grid points are used. If th ¡ 0 then all grid points are used. If th = large number, perhaps 10, then only the grid points on the 'coarsest' grid are used.
- Nd = Number of wavelet decompositions, e.g., if Nd = 3, then the ratio of the maximum Δx to the minimum Δx is $8 = 2^3$.
- iw = Width of wavelet refinement stencil. If $iw = 1$, then the magnitude of wavelet coefficients are checked at three locations from -iw to iw or at the locations -1, 0, 1 in order to determine if the grid point at location 0 should be used. So that if one has a hyperbolic system, or traveling waves, then if $iw > 2$ one can add grid points by looking 'backwards' and 'forwards' for a perturbation which might move into the region currently being examined. This is a kind of preparation for the future evolution of the system at hand.

Output Variables

- xo = The new wavelet-chosen grid. Note that the grid points on the boundaries are always used. That is, xo(1) = xi(1) and xo(No) = xi(N+1).
- fo = The function values on the new wavelet-chosen grid.
- No = The number of grid points in the new grid. Note that whereas 'No' counts every grid point, the input variable 'N' does not include the last point on the right hand boundary. This is done to facilitate the use with periodic as well as non-periodic boundary conditions.

4.2 The Four Subroutines

The following four subroutines provide a stand-alone 1 dimensional WOFD grid generation package.

4.2.1 Wavelet Analysis and Select Grid: newgr.f

This is the main subroutine which will be called by the user created driver program.

```
1 subroutine newgr(xi,fi,L,N,th,Nd,iw,xo,fo,No)
2 parameter(Nmax = 260, Lmax = 8, Ndmax = 8)
3 real xi(Nmax),fi(Nmax),xo(Nmax),fo(Nmax),h(Lmax),g(Lmax)
4 real HPF(Nmax/2+Lmax,Ndmax),data(Nmax),th,Extdata(Nmax+Lmax)
5 call getcoef(L,h,g)
6 do i = 1,Nmax
7 data(i) = fi(i)
8 enddo
9 do idecomp = 1, Nd
10 Ndim = N/(2**(idecomp-1))
11 call constext(data,Ndim,L,Extdata)
12 call filter(Extdata,h,g,Ndim,L,data,HPF(1,idecomp))
13 enddo
14 igrid = 0
15 do 10, ipnt = 1,N
16 if ( abs( mod(ipnt-1,2**(Nd))) .LT. .00001 ) then
```

```
17 igrid = igrid + 1
18 xo(igrid) = xi(ipnt)
19 fo(igrid) = fi(ipnt)
20 endif
21 do 20, idecomp = 1,Nd
22 n1 = abs(ipnt - 2**(idecomp-1) - 1)
23 n2 = 2**(idecomp)
24 if ( abs(mod(n1,n2)) .LT. .00001 ) then
25 index1 = 1+nint(real(n1)/real(n2))
26 iflagpoint = 0
27 do iwiden = -iw, iw
28 iindex = index1 + iwiden
29 if(iindex.LE.1.OR.iindex.GE.N/(2**(idecomp)))then
30 iindex = index1
31 endif
32 if (abs(HPF(iindex,idecomp)).GT.th)then
33 iflagpoint = iflagpoint + 1
34 endif
35 enddo
36 if (iflagpoint .GE. iw+1) then
37 igrid = igrid + 1
38 xo(igrid) = xi(ipnt)
39 fo(igrid) = fi(ipnt)
40 endif
41 endif
42 20 continue
43 10 continue
44 xo(igrid+1) = xi(N+1)
45 fo(igrid+1) = fi(N+1)
46 No = igrid+1
47 return
48 end
```

4.2.2 Get Daubechies Wavelet Coefficients: getcoef.f

This subroutine is called by *newgr.f* and its only function is to get the wavelet coefficients. Included here are the numbers only for the D_4 wavelet. Other wavelet coefficients can be added by the user. The numerical values for the coefficients h(:) in the following subroutine came from [1].

```
1 subroutine getcoef(L,h,g)
2 parameter (Lmax = 8)
3 real h(Lmax), g(Lmax)
4 h(1) = .482962913145
5 h(2) = .836516303738
6 h(3) = .224143868042
7 h(4) = -.129409522551
8 do i = 1,L
9 h(i) = h(i)/(sqrt(2.0))
10 enddo
11 do i = 1,L
12 g(i) = (-1)**(i-1) * h(L - i + 1)
13 enddo
14 return
15 end
```

4.2.3 Apply Wavelet Filter: filter.f

This subroutine does the actual wavelet filtering by dividing *Extdata* into its *high* and *low* components.

```
1 subroutine filter(Extdata,h,g,N,L,low,high)
2 parameter (Nmax = 260,Lmax = 8)
3 real low(Nmax/2+Lmax), high(Nmax/2+Lmax)
4 real Extdata(Nmax+Lmax), h(Lmax), g(Lmax)
5 do i = 1, Nmax/2+Lmax
6 low(i) = 0.0
7 high(i)= 0.0
8 enddo
9 do i = 1, N/2 + (L-2)/2
10 do j = 1,L
11 ij = 2*(i-1) + j - (L-2)
12 low(i) = low(i) + h(j) * Extdata(ij+2)
13 high(i) = high(i) + g(j) * Extdata(ij+2)
14 enddo
15 enddo
16 return
17 end
```

4.2.4 Apply Boundary Conditions: constext.f

This routine takes care of the boundary conditions by extending the scaling function coefficients in an appropriate way. The routine provided here extends with constant values. The user can define other routines for whatever boundary conditions are needed.

```
1 subroutine constext(data,N,L,Extdata)
2 parameter (Nmax = 260,Lmax = 8)
3 real data(Nmax), Extdata(Nmax+Lmax)
4 do i = 1, N
5 Extdata(L/2+i-1) = data(i)
6 enddo
7 do i = 1, L-3
8 Extdata(L/2-i) = data(1)
9 enddo
10 do i = 1, L-1
11 Extdata(N+L/2+i-1)=data(N)
12 enddo
13 return
14 end
```

5 Doubling the Grid Density

Note that the software included here takes a fine grid, V_0, and chooses from V_0 a subset of points from $W_1 \oplus W_2 \oplus W_3 \oplus V_3$ to obtain a numerical grid. It is possible that during a numerical simulation that even the finest grid Δx in V_0 is not fine enough and that a grid spacing $\Delta x/2$ is needed. This is possible by adding to V_0 the refinement W_0 to get $V_{-1} = V_0 \oplus W_0$. In the code this can be accomplished by first testing the magnitude of the wavelet coefficients in the subspace W_1 by adding a test statement similar to line 32 in which the magnitude of the numbers in HPF(:,1), corresponding to W_1, are tested against a second threshold number. For example, if th=.001 then one might decide to double the grid density if magnitude of HPF(:,1)> .01 (The reader should experiment with these numbers). If this test is true then exit *newgr.f* and double of the grid density of *xi* by interpolation thereby making N become $2N$ followed by another call to *newgr.f*. Note that it is necessary to double the grid before the data becomes too 'rough'. Once numerical oscillation has begun, it is too late. You must make your threshold numbers

sensitive enough to 'see' to high frequency regions coming and refine ahead of time. This type of intuition is easy to develop with a little practice.

6 2 Dimensional Examples

Applying this grid selection mechanism in higher dimensions is straightforward. For example, if one's data, perhaps pressure, is stored in a 2 dimensional array, then apply the 1 dimensional wavelet grid selection mechanism to each column and row. This mechanism will tell the user where grid points are needed. The user must then choose a set of grid points which contains the wavelet-generated set. Choosing a given subset of points from the set recommended by the wavelet analysis will depend on the application and the numerical method used, but if the set does not contain the wavelet-generated set of points, the user might find that there are features which are left unresolved.

The following subsections will illustrate the application of wavelets to choose grids. The first example comes from the original WOFD and the second example comes from an adaptive spectral method named WOFD2.

6.1 The Original WOFD

The original WOFD is spatially 4th-order accurate. The underlying grid at each level of refinement is a uniform grid, see Figures (1) and (2). Note that the grid has no dangling nodes. That is, differentiation occurs through any neighboring 5 points. At the boundary, the stencil is also five points but one sided. See [3] and [6] for details.

6.2 An Adaptive Spectral Method

The examples given in Figures (3) and (4) come from an adaptive spectral method named WOFD2, see [7]. WOFD2 adjusts the local grid density and the order of the local differentiation stencil based on wavelet analysis. The order of the differentiation operators could be as high as 32 or as low as 4. Orders above 4 require that the grid near the boundary be chebyshev in structure in order to implement high order boundary conditions. In other

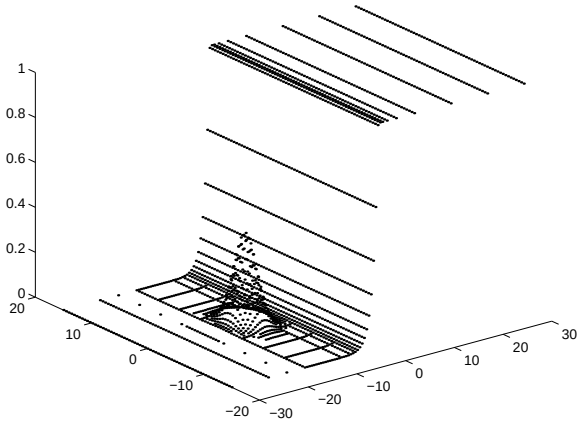

Figure 1: A domain which contains a high gradient flame front impinging on a Gaussian pulse.

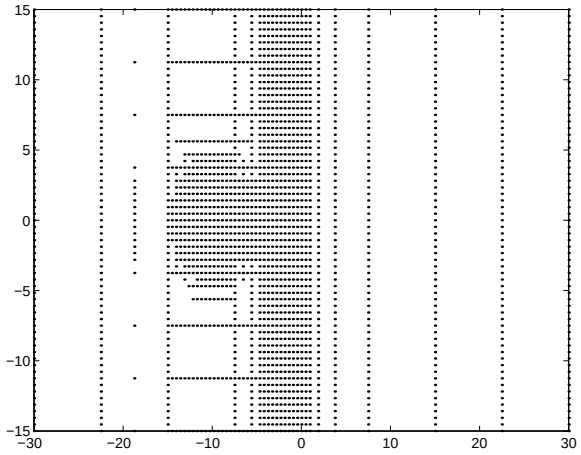

Figure 2: The grid selected by wavelets for a high gradient region impinging on a pulse.

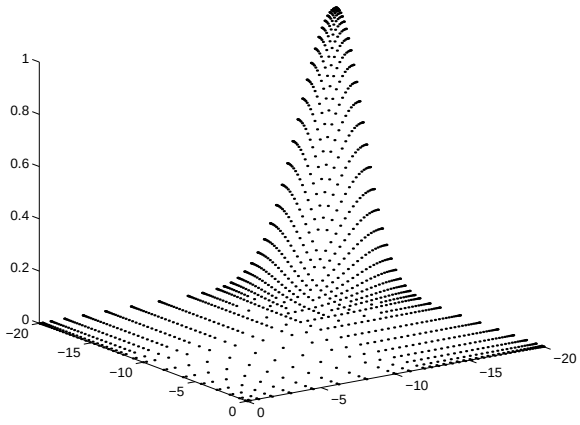

Figure 3: Gaussian pulse analyzed with wavelets in an adaptive spectral method.

words, if one wants to implement, say, 20th order boundary conditions then one must utilize a chebyshev grid near the boundary in order to control the Runge phenomenon which occurs with evenly-space grid distributions. To apply wavelet-based grid generation on Chebyshev grids, $x_i = \cos(\theta_i)$, then one applies the above based grid generation routine to the dependent variable f defined on the independent variable θ_i: $f(\theta_i)$. The following plots illustrate the application of WOFD2 in 2D to a Gaussian pulse entering the domain at one of the corners. Note that, as above, there are no dangling nodes. Again, the details of the application of wavelets to the adaptive spectral method WOFD2 can be found in [7].

6.3 Possible Applications to Finite Elements

The WOFD grid refinement mechanism and the software included in this paper can be applied to any flow in which one needs grid. The obvious extension would be to generate triangular grids for finite element analysis. In such an application one does not have an underlying fine grid at hand, but one can easily interpolate local data to obtain a local uniform grid. That is, wavelets are ideal for analyzing a domain and telling the user at what scale

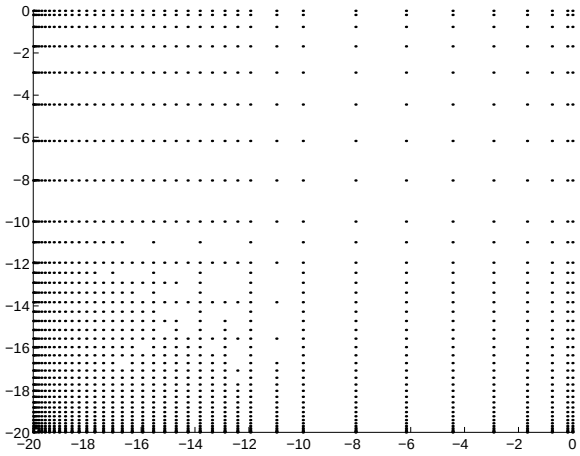

Figure 4: An adaptive Chebyshev grid generated by wavelets for an adaptive spectral method.

and where various information in the domain is contained. This is the type of information which is needed in order to choose grids, whether the grids are triangular or Cartesian.

7 Conclusion

Hopefully it has been illustrated within this paper that wavelets, with their ability to detect energy of flow variables at various scales and locations throughout a domain, provide a very natural mechanism for grid selection. In fact, this wavelet-based grid selection mechanism can be used to control the maximum error, L_∞ error, throughout the domain. One can simply set the parameter "th", as described above, to an acceptable error, and one will find at the end of the calculation that the L_∞ error will be of the same order of magnitude.

In addition, the underlying grid structure need not be uniform, as in Figure (2), but can be Chebyshev as in Figure (4), or any other grid structure as long as one can interpolate to a uniform grid for the grid selection mechanism. After the grid has been selected from the uniform set, one simply

interpolates back to their desired grid structure.

Finally, this paper contains Fortran source code. It appears that it takes a very long time for a researcher who is not familiar with wavelets to, first, learn the theory, and, second, write software which works. Hopefully, this software will prove useful.

References

[1] I. Daubechies, "Orthonormal Basis of Compactly Supported Wavelets", Comm. Pure Appl. Math., **41** (1988), pp. 909-996.

[2] G. Erlebacher, M. Y. Hussaini, L. Jameson, (eds.) "Wavelets: Theory and Applications", Oxford University Press, 1996.

[3] L. Jameson, "On The Wavelet Based Differentiation Matrix", Journal of Scientific Computing, September 1993 and ICASE Report No. 93-95, NASA CR-191583.

[4] L. Jameson, "On the spline-based wavelet differentiation matrix", ICASE Report No. 93-80, November 5, 1993, 37 pages. Applied Numerical Mathematics, Vol. 17, 1995, pp. 33-45.

[5] L. Jameson, "On The Differentiation Matrix for Daubechies-Based Wavelets on an Interval", SIAM J. Sci. Comp., March 1996 and ICASE Report No. 93-94, NASA CR-191582.

[6] L. Jameson, "On the Wavelet-Optimized Finite Difference Method", ICASE Report No. 94-9, NASA CR-191601.

[7] L. Jameson, "A Wavelet-Optimized, Very High Order Adaptive Grid and Order Numerical Method", ICASE Report No. 96-30, NASA CR-198331, May 3, 1996, 42 pages. Submitted to SIAM Scientific Computing.